How to / Maniaplanet / Manialink / Build a Manialink

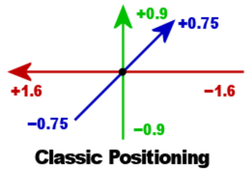
Author : Marcel @ Funtrackers

1.2 - Basic positioning of the ManiaLink elements

Before we finally get known the different elements of a ManiaLink, we still need some more basics: The positioning system. All visible elements can mostly be randomly placed on the screen, without having to worry about other elements. But without knowing the system, it is hard to place anything anywhere.

Better safe than sorry: The two positioning systems

Basically you have a full three dimensional coordinate system, with an X-axis for moving elements to the left and right, a Y-axis for moving them up and down, and because elements may overlap with some others, a Z-axis for moving elements in front and behind others. To make things a bit more complicated, you now have the choice between two different coordinate systems:
They both are as powerful as the other and you don't have any back draws using either of them, the only differences are the exact values of the axes and in which direction they are directed. Look at the following figures to exactly see these differences:



Classic Positioning

Menu Positioning

As you can see, converting between these two systems is very easy; you simply have to divide the values through or multiply with 100, paying special attention to the inverted X- and Z-axes. Nevertheless it is recommended to decide for one of these systems, and to use this system throughout the whole ManiaLink.

In fact, the difference in the ManiaLinks (except the values) is very small: If you want to use the Classic Positioning, you have to use the attributes pos and size, to use Menu Positioning add an additional n to get posn and sizen. Whenever an element is not placed where you expect it, always make sure you use the correct attributes and the correct orientation. Additionally, check to have version="1" specified in the opening <manialink> tag, as this, too, influences the values of the axes.

As already said, the Z value specifies which element is covered by which others. The bigger (menu system) or lower (classic system) the value is, the more in front it will appear. Be careful with these Z values: If two elements have the same values, or if an element has an invalid (too big or too low) value, the order of the elements will be mostly random, and it may even change if you for example move your mouse into the top bar of ManiaPlanet. So always make sure to have different Z values if elements overlap each other.

But why do we have two different coordinate systems, where one would clearly be enough? The reason is mostly historical: Where the Classic Positioning system was used back in the very first version of ManiaLinks, the United ManiaLinks, the newer Menu Positioning was introduced with the Forever ManiaLinks as a separate system, because the old system was not very good suited to be used in the game menus of TrackMania Forever.

Nevertheless, the old system was kept to correctly calculate positions of United ManiaLinks, and so they still both exists in ManiaPlanet ManiaLinks as of backwards compatibility. At least, to make things easier, the concrete values of both systems have been made more similar when 16:9 widescreen has been introduced into the ManiaLinks, so that converting between the systems now is no problem any longer.

Have a break: First examples of positioning

We are nearly finished with the basics, but before we continue with the last missing piece, the alignment of elements, I want to give you same examples to make the positioning of elements clear. In these examples, I will use Quads for displaying the positions, as a Quad we can simply assign a background color to make it visible. We will learn the different elements available for a ManiaLink later.

Let's start with a very basic example:

```
<quad bgcolor="F00A" />
```

We neither have specified a position, nor a size of the Quad. (Only the background color have been specified to make this Quad visible.) Nevertheless, we will get a Quad displayed with this code, because ManiaPlanet applies default values to all missing attributes. The following two lines will produce the exactly same Quad as the above one, with the only difference that we explicitly gave the position and the size:

```
<quad posn="0 0 0" sizen="100 100" bgcolor="F00A" />
<quad pos="0 0 0" size="1 1" bgcolor="F00A" />
```

We see two things on this example code:
First, as already mentioned above, the two positioning systems are equally powerful. Whatever we specify in the menu system (first Quad), we can convert to the classic system (second Quad). Because of this fact, the following examples and the rest of the Tutorial will always use the menu system, but of course you are free to use the classic one instead.
The second thing we see are the default values for these attributes: If the position is omitted, all coordinates are set to 0. If the size is omitted, it is set to 100 or 1 respectively for both width and height.

```
<quad posn="40 -40" sizen="20 20" bgcolor="00FA" />
```

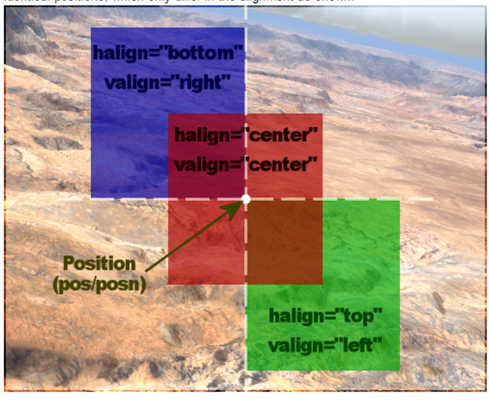
If we do not need to set the Z value to something different from 0, we can omit it and only specify to values for X and Y. We specified a position different from zero, so this Quad will not be in the middle of the screen anymore.
What do you think: In which direction has the Quad been moved by setting the position (in menu coordinates) to "40 -40"? Copy the line in a ManiaLink file and check your anwser.

```
<quad posn="-160 90 -75" sizen="320 180" bgcolor="0008" image="greytransparent.png" />
```

This last Quad is an example for inserting a background image into your ManiaLink: It covers the whole screen, and has been set to the very back using the lowest allowed value for Z (using the menu system)

The perfect confusion: The influences of halign and valign

Another thing which is important in context of the basic positioning of elements is their alignment, exactly their halign (horizontal alignment) and valign (vertical alignment). Let's have a look on the following screenshot showing three elements with identical positions, which only differ in the alignment as shown:



Without knowing the secret, the alignment seems to be confusing. The left aligned element is right of the right aligned one? And the bottom aligned one is above that one with top? The only thing you have to know is that you do not specify where the element should be, but where the position (specified with pos or posn) is.

So take another look on the screenshot, and look at where exactly the position is related to the elements. And now, the alignment starts to make sense: When we specify halign="bottom" and valign="right", the positioning point is at the bottom right corner, thus the element is above and right from it.

So always remember: halign and valign specify where the positioning point is, not where the element should go.

This site is friendly to stationary (PC/MAC) and mobile (Android/IOS) browsers
Trackmania, Shootmania, Questmania games & media © Nadeo & UbiSoft used under fair use policy
8,373,029 page views since January 2008. All tutorials, services, coding © Mania Creative 2008-2013